

Remarks

The above Amendments and these Remarks are in reply to the Office Action mailed January 6, 2009.

I. Summary of Examiner's Rejections

In the Office Action mailed January 6, 2009, Claims 1 and 18 were rejected under 35 U.S.C. 102(b) as being anticipated by Leymann et al. (U.S. Patent No. 6,308,224 B1, hereinafter Leymann).

Claims 2-3, 19 and 20 were rejected under 35 U.S.C. 103(a) as being unpatentable over Leymann in view of Meredith (U.S. 6,516,322 B1).

II. Summary of Applicant's Amendments

The present Reply amends Claims 1 and 18, and adds Claims 43-61, leaving for the Examiner's present consideration Claims 1-3, 18-20 and 43-61. Reconsideration of the Application, as amended, is respectfully requested.

III. Claim Rejections under 35 U.S.C §102(b)

In the Office Action mailed January 6, 2009, Claims 1 and 18 were rejected under 35 U.S.C. 102(b) as being anticipated by Leymann.

Claim 1

Claim 1 has been amended to more clearly define the embodiment therein. As amended, Claim 1 defines:

1. *A method for extending an existing object oriented programming language, comprising the steps of:*
 - selecting a program source file including a workflow definition created using a workflow language, wherein the program source file includes a source code and classes therein and a workflow definition created using the workflow language that is specified in the form of annotations to the source code and the classes;*
 - extending the source code with a plurality of workflow constructs, including an action construct representing an activity that allows a first software component to call an operation on a second software component; and*
 - using a workflow program according to the workflow definition, including processing, using a computer including a processing device operating thereon, the action construct to allow the first software component to call the operation on the second software component, and*

passing, according to the workflow definition in the form of annotations to the source code, information selected from one or more files, documents and/or tasks between system resource,s according to a set of procedural rules to generate activities at the computer as defined by the workflow definition.

Claim 1 has been amended to more clearly define selecting a program source file including a workflow definition created using a workflow language. The program source file includes a source code and classes therein and a workflow definition created using the workflow language that is specified in the form of annotations to the source code and the classes. The source code is extended with a plurality of workflow constructs, including an action construct representing an activity that allows a first software component to call an operation on a second software component. The workflow program is used to process the action construct to allow the first software component to call the operation on the second software component, and pass, according to the workflow definition in the form of annotations to the source code, information selected from one or more files, documents and/or tasks between system resources according to a set of procedural rules to generate activities at the computer as defined by the workflow definition.

The advantages of the embodiment defined by Claim 1 include that it provides for a workflow language in the form of annotations to the source code and the classes to enable developers that are used to using a specific programming language, to extend the programming language by adding workflow constructs that are missing from the programming language but desirable. For example, in accordance with an embodiment, developers that are familiar with Java can add workflow definitions using a workflow language in the form of annotations to the source code and the classes of Java source code.

In a traditional system, such annotation would normally be ignored when the source code is processed, because annotations normally are not part of the program itself, and they normally have no direct effect on the operation of the source code they annotate. However, in accordance with the embodiment defined by Claim 1, such annotations (e.g., created using a workflow language, and provided, e.g., as XML commands) are read by the system and are used to create a workflow program that passes information between system resources according to a set of procedural rules such that the system acts upon the information.

Leymann discloses a method of extending the specification of a process model within a workflow process environment (Abstract). An example of a process model, as disclosed in Leymann, is FLOWMARK, which supports the modeling of business processes as a network of activities (column 2, lines 13-14). The network of activities is referred to as the process model (column 2, lines 14-16). The method of extending the process model is accomplished by relating the process activity of the process

model to at least one object class and one object method residing within the object environment and implementing a process activity (column 3, lines 23-26).

Claim 1, as amended, defines *selecting a program source file including a workflow definition created using a workflow language, wherein the program source file includes a source code and classes therein and a workflow definition created using the workflow language that is specified in the form of annotations to the source code and the classes*. It was alleged in the Office Action that Leymann discloses this feature of Claim 1. Applicant respectfully submits that this feature is not disclosed in Leymann.

In particular, Claim 1 defines selecting a program source file that includes *a source code and classes*. In contrast, Leymann discloses selecting a workflow specification. It was asserted in the Office Action that a workflow specification is considered a program source file including a workflow definition created using the workflow language FLOWMARK definition language and that the workflow specification includes classes since the workflow specification is linked to classes. Applicant respectfully disagrees with this assertion. In Leymann, the workflow specification is linked to the class (column 3, lines 23-25). Claim 1 as amended instead defines selecting a program source file that includes a source code and classes. Thus, for the reasons stated above, selecting a workflow specification is not the same as selecting a program source file that includes a source code *and classes*.

Furthermore, Claim 1 defines *a workflow definition created using the workflow language that is specified in the form of annotations to the source code and the classes* that is used to extend the source code. It was alleged in the Office Action that Fig 19; column 3, lines 16-26; column 8, lines 4-61; and column 12, line 65-column 13, line 45 of Leymann disclose this feature of Claim 1. Applicant respectfully submits that this feature is not disclosed in Leymann.

In particular, Leymann discloses extending a workflow definition, created using the workflow language FLOWMARK, by linking the workflow definition with an object environment within which the process activity is to be implemented. However, although Leymann discloses extending a workflow specification, where object code is linked to a process activity to implement the process model, Leymann does not disclose that a workflow language is used to extend source code. Additionally, as mentioned above, a process model defines a business process (e.g., FLOWMARK is an example of one way to model a business process) and the specification of the process model is extended by linking the process model with an object environment within which the process activity is to be implemented. Claim 1 instead defines that source code is extended with a workflow construct. Applicant respectfully submits that linking a process activity of a process model to object code is not the same as *a workflow language*

that is specified in the form of annotations to the source code and the classes that is used to extend the source code, as defined in amended Claim 1.

Claim 1 further defines *extending the source code with a plurality of workflow constructs*. It was alleged in the Office Action that Leymann discloses this feature of Claim 1.

Applicant respectfully submits that this feature is not disclosed in Leymann. In particular, Leymann discloses that a process model is extended by linking the process model with an object environment within which the process activity is to be implemented. Claim 1 as amended instead defines that the source code is extended with a plurality of workflow constructs. As described above, a source file is not the same as a process model (i.e., workflow specification), and as such, extending a process model can not be the same as extending a source file as defined in Claim 1.

Additionally, as defined in Claim 1, the source code is extended using the workflow language that is specified in the form of annotations to the source code and classes. It was asserted in the Office Action that Leymann's use of keywords to relate a process model to an object code to invoke a program associated with a program activity (column 7, lines 48-49) discloses using annotations to extend source code, as defined in Claim 1. Applicant respectfully submits that extending source code with annotations (as defined in Claim 1) is not the same as using keywords to relate the process activity to an object class, as described in Leymann.

In particular, as disclosed in Leymann, keywords are not placed into object code (e.g., they are placed in process models). Nor do the keywords define a workflow (e.g., they link process activity to object code) and the keywords do not extend the source code to include a plurality of workflow constructs. Thus, using keywords to link process activity to object code as disclosed in Leymann is not a workflow definition created using a workflow language, as defined in Claim 1.

Claim 1 further defines extending the source code with constructs, *in the form of annotations to the source code and the classes*, where extending the existing object oriented programming language includes *passing, according to the workflow definition in the form of annotations to the source code, information including files, documents and/or tasks between system resources, according to a set of procedural rules to generate activities at the computer as defined by the workflow definition*. Applicant respectfully submits that Leymann does not disclose this limitation of Claim 1, as amended.

In view of these comments, Applicant respectfully submits that Claim 1 is not anticipated, nor otherwise rendered obvious, by the cited reference, and reconsideration thereof is respectfully requested.

Claim 18

The comments provided above with respect to Claim 1 are hereby incorporated by reference. Claim 18, while independently patentable, recites limitations that, similarly to those described above with respect to Claim 1, are not taught nor otherwise rendered obvious by the cited reference. Reconsideration thereof is respectfully requested.

Claims 2-3, 19 and 20

Claims 2-3, 19 and 20 depend from and include all of the features of Claims 1 or 18. Claims 2-3, 19 and 20 are not addressed separately but it is respectfully submitted that these claims are allowable as depending from an allowable independent claim, and further in view of the comments provided above.

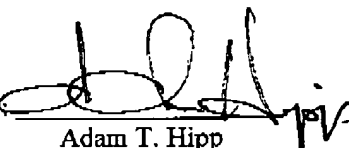
It is also submitted that these claims add their own limitations which render them patentable in their own right. Reconsideration thereof is respectfully requested.

IV. Conclusion

In view of the above amendments and remarks, it is respectfully submitted that all of the claims now pending in the subject patent application should be allowable, and reconsideration thereof is respectfully requested. The Examiner is respectfully requested to telephone the undersigned if he can assist in any way in expediting issuance of a patent.

The Commissioner is authorized to charge any underpayment or credit any overpayment to Deposit Account No. 06-1325 for any matter in connection with this reply, including any fee for extension of time, which may be required.

Respectfully submitted,

By: 
Adam T. Hipp
Reg. No. 60,334

Date: April 6, 2009

Customer No. 23910
FLIESLER MEYER LLP
650 California Street, 14th Floor
San Francisco, California 94108
Telephone: (415) 362-3800

- 11 -

Application No. 10/784,374
Attorney Docket No.: ORACL-01389US2
M:\ahipp\wp\ORACL\1389US2\1389US2_ROA_010609_2.doc